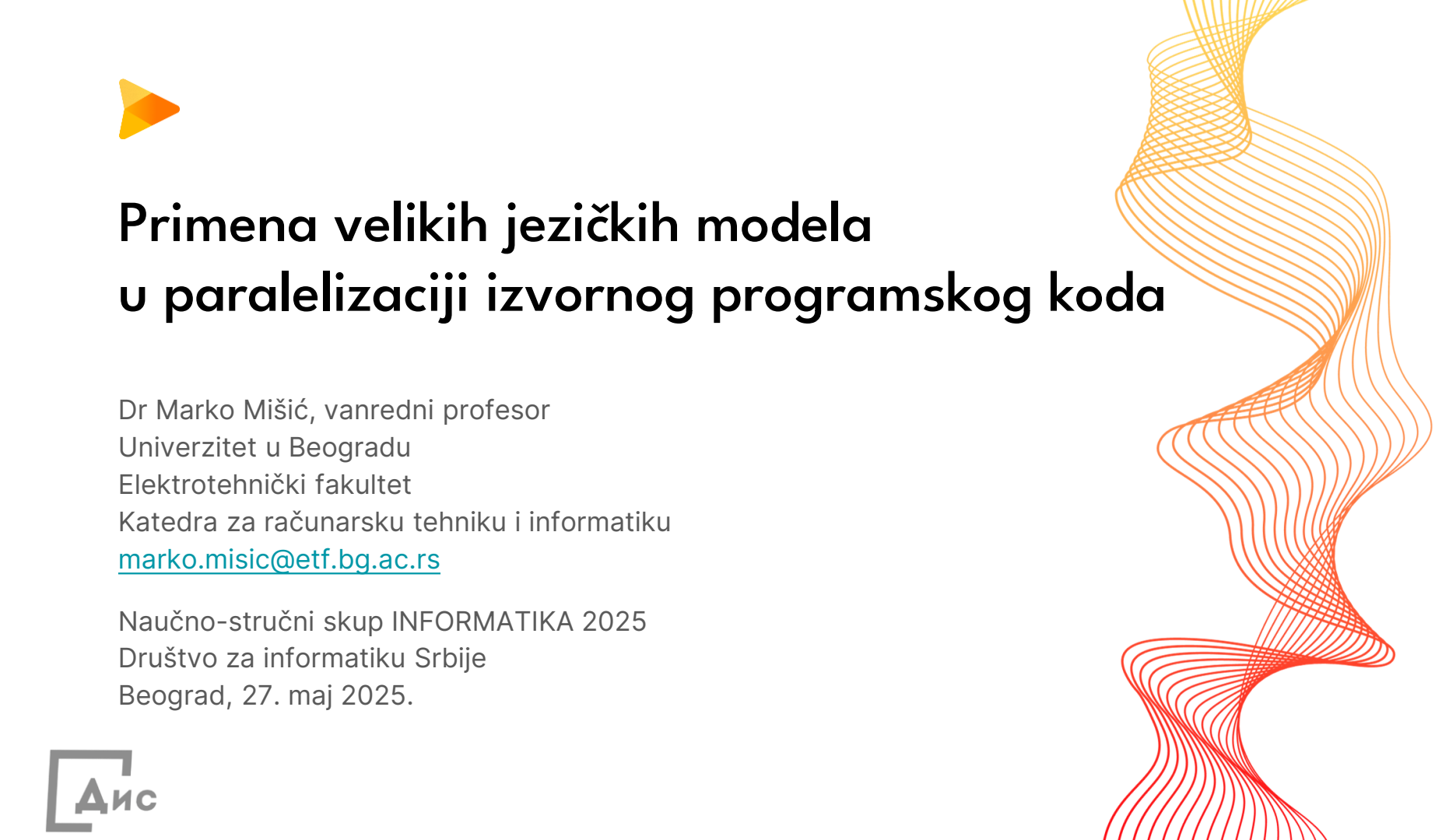


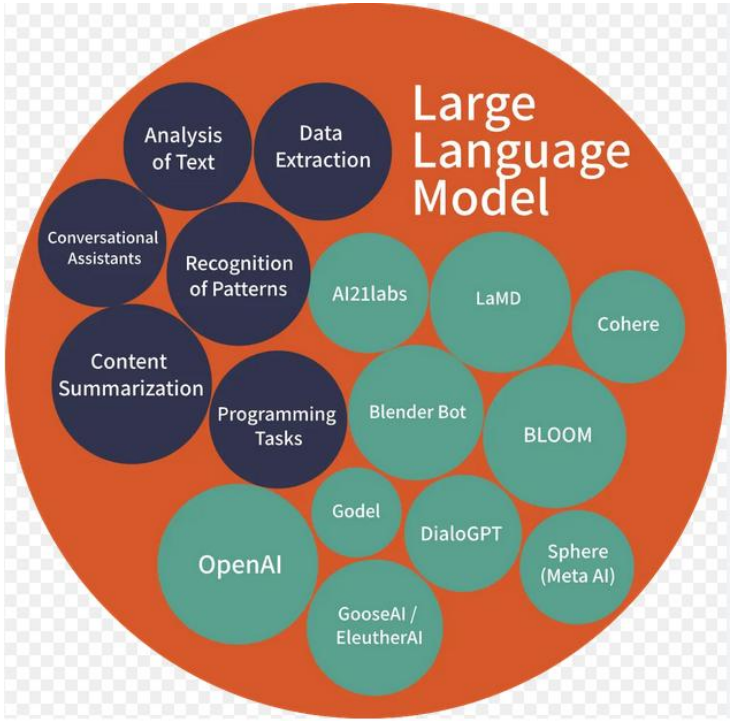
Primena velikih jezičkih modela u paralelizaciji izvornog programskog koda

Dr Marko Mišić, vanredni profesor
Univerzitet u Beogradu
Elektrotehnički fakultet
Katedra za računarsku tehniku i informatiku
marko.misic@etf.bg.ac.rs

Naučno-stručni skup INFORMATIKA 2025
Društvo za informatiku Srbije
Beograd, 27. maj 2025.

Uvod

- **Veliki jezički modeli** (Large Language Models - LLMs) se koriste u velikom broju domena u proteklih nekoliko godina
  - Korisnička podrška, marketing, finansije, društvene nauke, mašinsko prevođenje, generisanje sadržaja, ali i u **programiranju**
- LLMs iskazuju ozbiljne mogućnosti u rešavanju **složenih programerskih i računskih problema**
- **Računarstvo visokih performansi** (High-performance computing - HPC) teži obradi velikih količina podataka u što kraćem vremenu korišćenjem **paralelizma** koji je dostupan u hardveru
  - Zahteva korišćenje paralenog programiranja i paralelnih programskih modela



Veliki jezički modeli (1)

- Veliki jezički modeli su **probabilistički** modeli zasnovani na **prirodnim** jezicima
  - Osnova su za današnje **generativne** modele
- Veći broj veoma **moćnih** modela
  - OpenAI **GPT-4** & **GPT-4 Turbo**
  - Google DeepMind – **Gemini** 1.5 serija
  - Anthropic – **Claude** 3 serija
  - Mistral – **Mistral** 7B & Mixtral
  - xAI – **Grok**
  - Meta **LLaMA** 2 & LLaMA 3 serija
  - DeepSeek VL & **Coder** & LLM

Veliki jezički modeli (2)

- **Generisanje koda** je jedna od **fokalnih** aplikacija danas sa primenama kao što su:
  - **Razumevanje** koda i **generisanje izveštaja**
  - **Recenzija** programskog koda
  - **Popravka** i **refaktorisanje** programskog koda
  - Jednostavniji **programski** zadaci
  - Primena u edukaciji (studentski **domaći** i **projektni** zadaci)
- Alati **ChatGPT**, Github **Copilot**, Continue, CodeWhisperer, Gemini
  - U obliku *plugin* dodataka, *chat* klijenata i **CLI/terminal** alata
  - Visoke **tačnosti >80%** na *benchmark* skupovima poput **HumanEval**

Paralelizacija programskog koda

- Veliki broj **paralelnih** programskih modela
  - **Deljena memorija** – procesi, niti (Pthreads, OpenMP, C++ *threads*, Java...)
  - **Distribuirana memorija** (MPI, GASPI, Julia...)
  - **GPGPU** (CUDA, OpenCL, HIP, OpenACC, OpenMP...)
  - **Hibridni** modeli (Kokkos, Raja, oneAPI, SYCL...)
- Pisanje paralelnih programa je **izazovan** zadatak
  - Pronalaženje **paralelizma** u problemu koji se rešava, korišćeni programski **model**, **deljenje** podataka, **sinhronizacija** i **komunikacija** između paralelnih zadataka...
  - Održavanje **baze** koda za različite **platforme**

OpenMP paralelni programski model

- Programski model **deljene memorije** zasnovan na **direktivama**
  - Asistirana, **implicitna** paralelizacija programskog koda
  - Programer prepoznaje **izvore paralelizma**, ali i potencijalne **zavisnosti** po podacima
  - Podrška za **CPU** i **akceleratore** (od standarda 4.0)
- Programer **anotira** kod direktivama
  - Radi nad postojećim, **sekvencijalnim** kodom
  - **Održava** istu ili sličnu bazu programskog koda
- **Prevodilac** proizvodi **paralelni kod** na osnovu **direktiva**

OpenMP

```
#pragma omp parallel default(shared) \\
                     private(i)
{
 #pragma omp for reduction(+:sum)
   for(i = 0; i < n; i++)
     sum += i;
}
```

Дис

Eksperiment

- **Evaluacija** mogućnosti ChatGPT i Github Copilot alata da proizvedu OpenMP paralelni programski kod zasnovan na postojećem sekvencijalnom kodu
- Devet **mini-aplikacija** sa različitim šablonima izvršavanja
- **Četiri različite verzije** za svaku mini-aplikaciju
  - Osnovna, sekvencijalna verzija (***base***)
  - Ručno urađena OpenMP verzija (***omp***)
  - Po dve verzije produkovane od strane ChatGPT (***omp_gpt***, ***omp_gpt2***) i Github Copitlot (***copilot***)
    - **Delimični** i **kompletan** kontekst koda
    - **"Try to parallelize the code snippet given below using OpenMP"** sa podešavanjima

Mini-aplikacije

- Uglavnom iz **naučnog** domena
  - **Numeričke** metode, simulacije iz domena **računske fizike, hemije**

Application	Source	Type	Description
Feynman	Burkardt	Compute bound	Feynman-Kac algorithm to solve Poisson's equation in 3D interval
HotSpot	Rodinia	Memory bound	Thermal simulation
Mandelbrot	EPCC Training Examples	Compute bound	Calculates an area of the Mandelbrot set
MolDyn	EPCC Training Examples	Compute bound	Molecular dynamics simulation
Nbody	ITPP Book	Compute bound	N-body interaction simulation
PiCalculation	ITPP Book	Compute bound	Estimates the value of PI
Prime	Burkardt	Compute bound	Calculates prime numbers
Saxpy	AMD HPC Training Examples	Memory bound	Scalar multiplication and vector addition operation
Sgemm	Parboil	Memory bound	Single precision dense matrix-matrix multiplication kernel

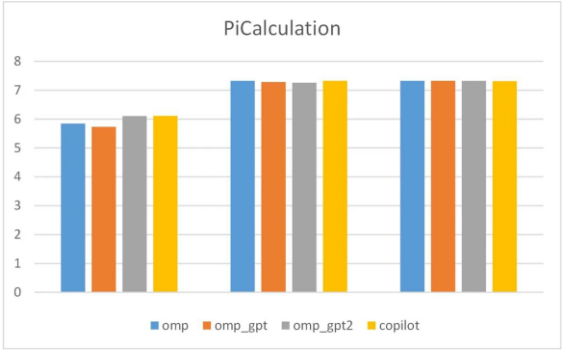
Rezultati – primer tri mini-aplikacije

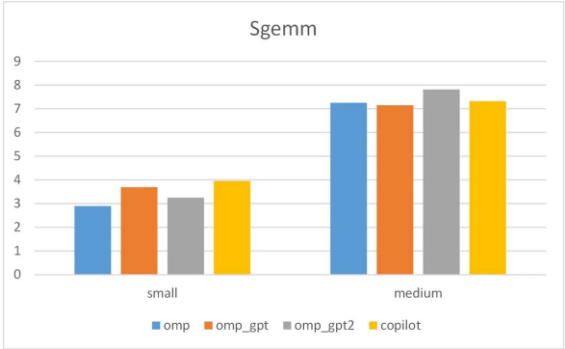
- PI – Oba alata korektno anotiraju **glavnu petlju** algoritma i deklarišu **redukcionu** promenljivu
- SGEMM – Oba alata korektno anotiraju **dve ugneždene petlje** i umeću *collapse* odredbu
  - ChatGPT ubacuje suvišnu *atomic* direktivu za sinhronizaciju i nepotrebnu *firstprivate* clause
- MANDELBROT – Problemi sa složenim **strukturama podataka**, **sinhronizacijom**, *collapse* odredbom, **raspoređivanje** iteracija/poslova
- Statistika interakcija sa alatima
  - **Broj upita** za svaku mini-aplikaciju pre prvog uspešnog **prevođenja**, prvog uspešnog **izvršavanja** i optimalnog izvršavanja

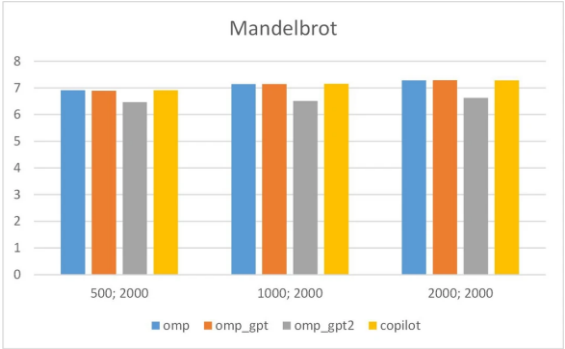
Application	Compilation success		Correct execution		Optimal execution		
	omp_gpt	omp_gpt2	omp_gpt	omp_gpt2	omp_gpt	omp_gpt2	copilot
PiCalculation	1	1	1	1	1	1	1
Sgemm	1	1	2	1	3	2	2
Mandelbrot	2	1	2	1	7	3	1

Rezultati analize

- **Slične performanse** nakon optimizacije
- Uočeni **problemi**
  - Određivanje opsega vidljivosti promenljivih
  - Nepotrebna privatizacija promenljivih
  - Suvišna sinhronizacija
  - Nepoznavanje strategija raspoređivanja
  - Rukovanje složenim strukturama podataka
  - Razumevanje konteksta programskog koda
- Potrebno strpljenje prilikom zadavanja dodatnih upita
  - Unapređivanje *prompt engineering* tehnika!







Zaključak

- Oba **alata** su u stanju da korektno anotiraju kod **u većini slučajeva**
- Alati još uvek **ne mogu** u potpunosti da dostignu **performanse** ručno napisanog i optimizovanog koda u svim slučajevima
  - Ali skoro da smo tamo stigli!
- Modelima nedostaje **dublje razumevanje** različitih aspekata paralelnog programiranja
  - **Sinhronizacija** i **strategije raspoređivanja** poslova predstavljaju potencijalne probleme
- **Budući rad** uključuje:
  - Testiranje **specifičnih model**a treniranih na **HPC** skupovima podataka (OMPify, AutoParLLM, OMPGPT...)
  - Više **raznovrsnijih** mini-aplikacija – NPB, SPEC, Rodinia, Parboil
  - Različite *prompt engineering* tehnike

Reference

- **M. Mišić**, M. Dodović, An assessment of large language models for OpenMP-based code parallelization: a user perspective, Journal of Big Data, Vol. 11, No. 161, Nov, 2024
- **M. Mišić**, M. Dodović, Can ChatGPT write parallel code?, 3rd Serbian International Conference on Applied Artificial Intelligence (SICAAI), Univerzitet u Kragujevcu, Kragujevac, May, 2024
- Đ. Pešić, M. Vujošević Janičić, **M. Mišić**, J. Protić, Assessing ChatGPT for Algorithm Time Complexity Education, Foundations of Computer Science and Frontiers in Education: Computer Science and Computer Engineering (CSCE 2024), Communications in Computer and Information Science, pp. 182 - 191, Springer, Cham, Las Vegas, NV, USA, Apr, 2025
- Đ. Pešić, M. Vujošević Janičić, **M. Mišić**, J. Protić, Generisanje programskih segmenata zadate složenosti pomoću alata veštačke inteligencije, Zbornik radova 30. IKT konferencije "YU INFO 2024", pp. 74 - 79, Informaciono društvo Srbije, Kopaonik, Srbija, Mar, 2024

Hvala na pažnji! 😊

Primena velikih jezičkih modela u paralelizaciji izvornog programskog koda

Dr Marko Mišić (marko.misic@etf.bg.ac.rs)

Naučno-stručni skup INFORMATIKA 2025
Društvo za informatiku Srbije
Beograd, 27. maj 2025.